PL Assignment #2 : Recognizing Tokens

과제물 부과일 : 2017-03-16(목)
Program Upload 마감일 : 2017-03-22(수) 23:59:59

문제

다양한 형태의 identifier, integer number(**음수 포함**) 들로 이루어진 문자열을 입력 받아, 각 요소를 인식하여 출력하는 program을 작성하시오.
예를 들어 입력 문자열의 내용이 아래와 같다면,

```
banana  267  h  cat  -3789  7  y2010
```

출력은 아래와 같아야 한다.

```
[[ID: banana], [INT: 267], [ID: h], [ID: cat], [INT: -3789], [INT: 7], [ID: y2010]]
[ID: banana]
[INT: 267]
[ID: h]
[ID: cat]
[INT: -3789]
[INT: 7]
[ID: y2010]
end
```

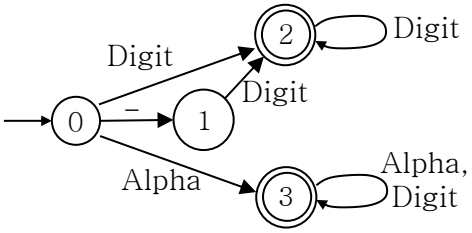
Regular Expression : 특정한 규칙을 가진 문자열의 집합

```
id:     Alpha[Alpha|Digit]*
int:    Digit+ | "-" Digit+
Alpha:  [A-Z] | [a-z]
Digit:  [0-9]
```

결정적 유한 오토마타

오토마타는 수학적 방법론에 바탕을 둔 프로그램의 추상적인 모델이다. 오토마타는 입력데이터를 읽을 수 있고, 결과를 accept, reject와 같은 특정 형태의 출력 기능을 가지고 있다. 결정적 유한 오토마타는 상태의 개수가 유한하고, 한 번에 하나의 상태로 전이한다.

mDFA



Programming

Token 표현하기

```
class TokenType():
    ID=3
    INT=2
```

Data Type

```
class Token():
    def __init__(self, type, lexeme):
        self.type=type
        self.lexeme=lexeme

    def __str__(self):
        return "[" + TOKENTYPE_NAMES[self.type] + ": " + self.lexeme + "]"
    def __repr__(self):
        return self.__str__()
```

Programming

```
class CuteScanner():

    def __init__(self, source):
        source=source.strip()
        #tokenize a string, delimiter is " "
        token_list=source.split(" ")
        #iterator
        self.token_iter=iter(token_list)


    def next_token(self):
        #문자열 하나를 받고, 토큰화함
        #토큰 타입을 분류한 뒤, 토큰 객체를 반환
        #상태 0에서 시작
        state=0

        #next()는 이터레이터로 부터 다음 item을 받아옴
        #더 이상 받을 것이 없으면 에러가 발생
        #두 번째 파라미터로 None을 넣음으로써, 더 이상 받을 item이 없으면 None을 받음

        #get if token exist
        temp_token= next(self.token_iter, None)

        if temp_token is None : return None
        #Python에서 문자열은 char의 List임
        for temp_char in temp_token:
            """:type : str"""
            if state==0:
```

```
            #읽은 문자가 숫자라면 상태를 2로 바꿈
            if temp_char.isdigit(): state=2
            #읽은 문자가 '-' 라면, 상태 1로 바꿈
            elif temp_char=='-': state=1
            #읽은 문자가 알파벳이면, 상태 3로 바꿈
            elif temp_char.isalpha():state=3
            else :
                print "ERROR"
                return None
        elif state==1:
            if temp_char.isdigit():
                #fill out if state is 1
            else :
                 print "ERROR"
                return None
        elif state==2:
            #fill out if state is 2

        elif state==3:
            #fill out if state is 3

        else:
            print "ERROR"
            return None

    if state==2:
        return Token(TokenType.INT, temp_token)
    elif state==3:
        return Token(TokenType.ID, temp_token)

def tokenize(self):
    #Type is List
    tokens=[]
    while True:
        #fill out
        #next_token()함수를 이용하여 토큰을 받고, tokens에 추가
        #python List 찾아서 할 것
    return tokens
```

```
def Test_CuteScanner():
    test_cute = CuteScanner("banana 267 h cat -3789 7 y2010")
    test_tokens=test_cute.tokenize()

    print test_tokens
    for token_i in test_tokens:
        print token_i
    print "end"

Test_CuteScanner()
```

유의 사항

- 주어진 Test함수를 수정하지 않는다.

수정: 2017-03-13